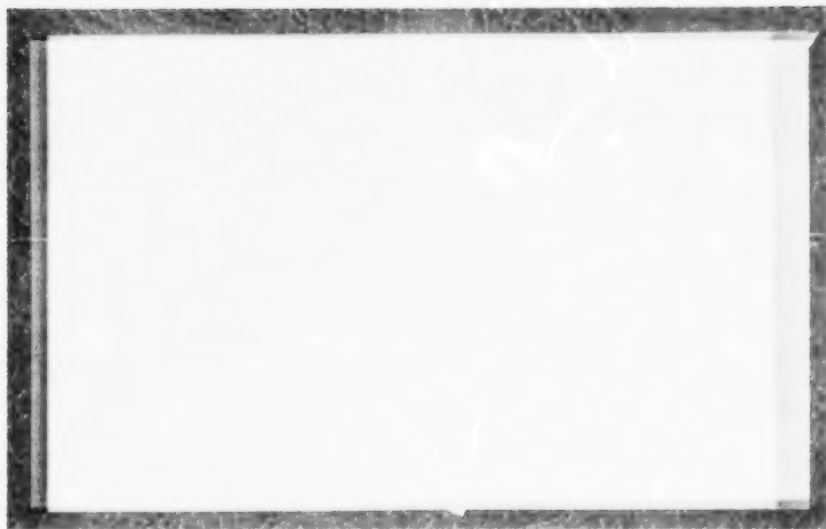


UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS



CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

VEHICLES SCHEDULING WITH SLIDING
TIME WINDOWS

by

Jacques A. Ferland*
and
Luc Fortin*

* Département d'informatique et de recherche opérationnelle.

The research for this paper was supported by NSERC (Grant A8312) and FCAR (Grant 0899). The second author was holding a postgraduate scholarship from NSERC to complete this research.

Cette étude a été publiée grâce à une subvention du fonds FCAR pour l'aide et le soutien à la recherche.

Université de Montréal

Centre de recherche sur les transports - Publication #515
Département d'informatique et
recherche opérationnelle - Publication #593

March 1987

Dépôt légal - 2e trimestre 1987

ABSTRACT

In this paper we introduce a heuristic approach for the vehicle scheduling problem with sliding time windows. In the problem with time windows, we take advantage of the fact that the starting time of each task must fall within a given time interval (i.e., a time window) rather than being fixed. Here, another level of flexibility is added by allowing the time windows of subsets of tasks to slide. The approach consists of identifying pairs of tasks offering good opportunity costs for reducing the overall cost, and finding ways of modifying the starting times to allow them to be linked. The technique is first developed for the problem with time windows and then extended to the case with sliding windows. Numerical results are reported to indicate the efficiency of the approach.

RESUME

Dans cette publication nous présentons une approche heuristique pour traiter le problème d'horaires de véhicules avec fenêtres de temps glissables. Dans le problème avec fenêtres de temps nous tirons avantage du fait que les débuts de tâches doivent se situer à l'intérieur d'intervalles (fenêtres de temps) plutôt que d'être des moments fixes. Ici, en permettant aux fenêtres de glisser, un nouveau niveau de flexibilité est ajouté. L'approche consiste à identifier des paires de tâches offrant un bon coût d'opportunité pour réduire le coût total de même que les modifications à apporter aux moments de début des tâches pour leur permettre de se suivre. La technique est d'abord développée pour le problème avec fenêtres de temps et ensuite étendue au cas où elles peuvent glisser. Des résultats numériques indiquant les performances de l'approche sont rapportés.

1. Introduction

In this paper we introduce a heuristic approach to the vehicle scheduling problem in which the time windows of subsets of tasks are allowed to slide. The approach relies heavily on the notion of the "opportunity cost" of linking two tasks to reduce the overall cost.

A general vehicle scheduling problem can be specified in terms of a set of tasks to be executed with a fleet of vehicles available. Each task i ($1 \leq i \leq n$) includes the following time and space (location) characteristics:

SL_i , the starting location

EL_i , the ending location

ST_i , the starting time

D_i , the duration

$ET_i = ST_i + D_i$, the ending time.

A vehicle executing task i must be at location SL_i at time ST_i . Then at time ET_i , the vehicle is located at EL_i and is available to execute other tasks. The problem is to partition the set of tasks into sequences of tasks each of which corresponds to the schedule of a vehicle. Hence no two tasks of a sequence can take place simultaneously, and task j can be executed after task i if, taking into account the time t_{ij} required for a vehicle to go from EL_i to SL_j , the vehicle can be made available at time ST_j at SL_j ; i.e.

$$ET_i + t_{ij} \leq ST_j.$$

In the vehicle scheduling problem with fixed starting times, time ST_i is fixed for each task i . In many realistic contexts, each task i can be initiated within a time interval $[a_i, b_i]$ referred to as a time window. In this case we have a vehicle scheduling problem with time windows. Now, suppose that the set of tasks T is partitioned into K subsets, $K \leq n$; i.e.

$$T = T_1 \cup T_2 \cup \dots \cup T_k.$$

Furthermore, all tasks $i \in T_k$ have the set starting time $\bar{a}_k$ in common as follows:

$$a_i = \bar{a}_k + f_i \text{ for all } i \in T_k.$$

The vehicle scheduling problem with sliding time windows is characterized by the fact that the set starting time $\bar{a}_k$ must be within a time interval $[\bar{a}_k^1, \bar{a}_k^2]$ (i.e., $\bar{a}_k \in [\bar{a}_k^1, \bar{a}_k^2]$) for all subsets T_k . In other words, even more flexibility is allowed in the problem because the time windows themselves can be adjusted (slid) to the left or the right, but all the tasks of the set are influenced accordingly.

Such a problem arises in the context of school busing. The routes are the tasks in this problem. In the afternoon, the students have to leave within some time interval after the school ending time, and time windows are specified accordingly. Furthermore, school schedules can be modified if this is worthwhile to reduce the fleet. But this has an impact on the time windows of all the routes leaving from that school. In [1, 2] the authors sequentially treat the school scheduling problem and the bus scheduling problem. The approach proposed here allows both problems to be handled simultaneously and should induce a reduction in fleet size.

The notion of the opportunity cost of linking two tasks by modifying their starting times is introduced in Section 2. Since we need to refer to the solution procedure for solving the vehicle scheduling problem with fixed starting times, this problem is reviewed first. Then three different ways of specifying the opportunity costs are derived. The notion is used extensively in the other sections. The approach is first introduced in Section 3 to handle problems with time windows where starting times are modified within the time windows. Then, in Section 4, it is extended to the problem where time windows can slide to the left or the right but

under the restrictions mentioned above. Finally, the paper concludes with Section 5 where numerical results are reported to indicate the efficiency of the procedure in reducing total cost measured in terms of the number of vehicles required.

2. Opportunity cost

First we review the problem with fixed starting times to introduce the notion of the opportunity cost of linking two tasks.

2.1. Vehicule scheduling with fixed starting times

A mathematical formulation of the problem with fixed starting times is derived using the precedence graph (N, A) of the tasks characterized as follows:

- i) With each task i associate two nodes o_i and $d_i \in N$ (o_i corresponding to the starting time of task i and d_i to its end). A node $g \in N$ is associated with the garage.
- ii) An arc (d_i, o_j) exists (i.e., $(d_i, o_j) \in A$ or task j can follow task i) if

$$ET_i + t_{ij} \leq ST_j .$$

Furthermore, for each task i , introduce arcs (g, o_i) and (d_i, g) allowing i to be the first or the last task of a sequence, respectively. (Note that no arc of type (o_i, d_j) , (g, d_j) , or (o_i, g) exists in this graph.)

Such a graph with 7 tasks is illustrated in Figure 2.1. In this example tasks 3, 4, 5, 6 and 7 can follow task 2, task 6 can follow task 4, and task 7 can follow task 5.

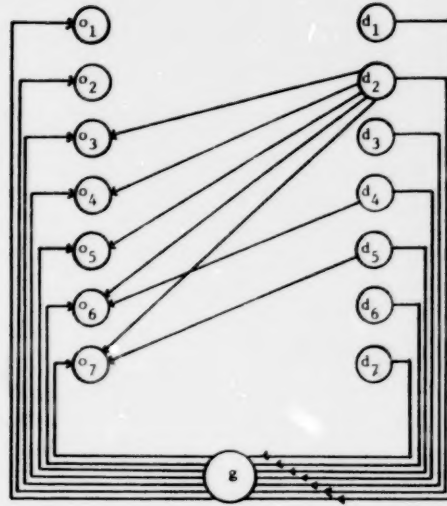


FIGURE 2.1

Precedence graph (N, A)

The problem can be formulated as a minimum cost flow problem defined on this precedence graph:

$$\begin{aligned}
 & \text{Min} \quad \sum_{(i, j) \in A} c_{ij} x_{ij} \\
 (FP) \quad & \text{Subject to} \quad \sum_{j \in P_{d_i}} x_{d_i j} = 1 \quad 1 \leq i \leq n \\
 & \quad \quad \quad \sum_{j \in B_{o_i}} x_{j o_i} = 1 \quad 1 \leq i \leq n \\
 & \quad \quad \quad \sum_{i=1}^n (x_{g o_i} - x_{d_i g}) = 0 \\
 & \quad \quad \quad 0 \leq x_{ij} \leq 1 \quad (i, j) \in A
 \end{aligned}$$

where $P_{d_i} = \{j \in N: \text{arc } (d_i, j) \in A\}$ and $B_{o_i} = \{j \in N: \text{arc } (j, o_i) \in A\}$. Given an optimal solution of this problem,

- i) if $x_{go_i} = 1$, then task i is the first of a sequence;
- ii) if $x_{dg} = 1$, then task i is the last of a sequence;
- iii) if $x_{d_o_j} = 1$, then task j follows task i in a sequence.

The linear costs c_{ij} are specified according to the objective of the problem. For instance, if the fleet size and the deadheading times are to be minimized, then

$$c_{d_o_j} = t_{ij} \quad \text{if } (d_i, o_j) \in A$$

$$c_{go_i} = c_{dg} = cg \quad \text{for } 1 \leq i \leq n$$

where the magnitude of cg relative to the t_{ij} 's measures the importance of fleet size versus deadheading times ($cg \gg t_{ij}$ in general).

Several specialized procedures exist to find an optimal solution of (FP) (see [7, Section 2]). But it can also be solved with a primal simplex algorithm specialized to the minimum cost flow problem [8, 11]. Such an algorithm will also generate simplex multipliers associated with the constraints of the problem. It is well known that if π_i denotes the simplex multiplier associated with node $i \in N$, then the relative cost of an arc $(i, j) \in A$ is given by the relation

$$\bar{c}_{ij} = c_{ij} - \pi_i + \pi_j. \quad (2.1)$$

We shall use this result to specify the notion of the opportunity cost of linking two tasks.

2.2. Opportunity cost

Suppose that given the starting times as specified, task j cannot follow task i (i.e., $ET_i + t_{ij} > ST_j$, and $\text{arc } (d_i, o_j) \notin A$). To predict the influence on the value of the objective function of modifying ET_i or ST_j to induce the possibility of linking task j to task i , we introduce the notion of the " opportunity cost " of introducing arc (d_i, o_j) in A .

Assume that an optimal solution for (FP) is available together with optimal multipliers associated with nodes in N . The " opportunity cost " of arc (d_i, o_j) is defined as

$$\bar{c}_{d_i o_j} = c_{d_i o_j} - \pi_{d_i} + \pi_{o_j} = t_{ij} - \pi_{d_i} + \pi_{o_j} \quad (2.2)$$

Note that if $\text{arc } (d_i, o_j) \in A$, then the opportunity cost is the relative cost of the arc (taking a nonnegative value at the optimum). If $(d_i, o_j) \notin A$ and $\bar{c}_{d_i o_j} > 0$, then relatively to the actual optimal basis, the objective function of (FP) cannot be reduced even if task j could be linked to task i . But, if $\bar{c}_{d_i o_j} < 0$, then the objective function might decrease if the link between them is activated. Indeed, assume that arc (d_i, o_j) could be introduced into A in such a way that the basic optimal solution on hand remains a basic feasible solution for the new problem (FP) where the non-basic variable $x_{d_i o_j}$ has a negative relative cost. Even if this variable becomes basic, it does not necessarily imply a decrease of the objective function because of the degeneracy inherent in any network flow problem. Nevertheless it is interesting to analyze the opportunity of linking task j to i if the opportunity cost is such that $\bar{c}_{d_i o_j} < 0$.

It is interesting to consider an alternate way of specifying the opportunity cost. Indeed, using the structure of the problem illustrated in Figure 2.1, it is easy to specify opportunity costs in terms of deadheading costs t_{ij} and vehicle cost c_g .

Assume that an optimal solution to (FP) (or a set of optimal sequences of tasks) is available. It is easy to generate a basic optimal solution of (FP) and the associated multipliers. First we illustrate the technique using the example in Figure 2.1. Assume that an optimal solution includes 4 sequences:

Sequence 1 : {1}

Sequence 2 : {2, 5, 7}

Sequence 3 : {4, 6}

Sequence 4 : {3} .

A corresponding basic optimal solution of (FP) is illustrated in Figure 2.2

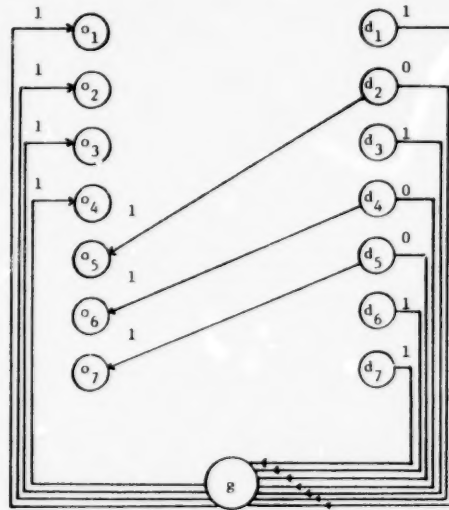


FIGURE 2.2

Basic optimal solution of (FP)

The variables

$$x_{go_1} = x_{go_2} = x_{go_3} = x_{go_4} = x_{d_1g} = x_{d_2g} = x_{d_3g} = x_{d_4g} = x_{d_5g} = x_{d_6g} = x_{d_7g} = x_{d_2o_3} = x_{d_5o_7} = x_{d_4o_6} = 1$$

are taken as basic variables. To complete the set of basic variables we use variables

$$x_{d_2g} = x_{d_4g} = x_{d_5g} = 0.$$

Hence, in general, it is easy to see that a basic optimal solution of (FP) is generated using the variables x_{dg} , $1 \leq i \leq n$, x_{go_i} for all tasks i that are the first in any sequence, and $x_{d_o_j}$ if task j is linked to i .

To evaluate the associated simplex multipliers, take $\pi_g = 0$. Since x_{dg} is basic for all $1 \leq i \leq n$, then for all nodes d_i

$$0 = \bar{c}_{dg} = c_{dg} - \pi_{d_i} + \pi_g = cg - \pi_{d_i},$$

and for $1 \leq i \leq n$

$$\pi_{d_i} = cg.$$

For any task i which is the first of some sequence, x_{go_i} is basic, and

$$0 = \bar{c}_{go_i} = c_{go_i} - \pi_g + \pi_{o_i} = cg + \pi_{o_i}.$$

Then

$$\pi_{o_i} = -cg.$$

For the other tasks, if task j follows task i , $x_{d_o_j}$ is basic, and

$$0 = \bar{c}_{d_o_j} = c_{d_o_j} - \pi_{d_i} + \pi_{o_j} = t_{ij} - cg + \pi_{o_j}.$$

Then

$$\pi_{o_j} = cg - t_{ij}.$$

Now, interesting interpretations of the opportunity cost are obtained with these multipliers. Indeed,

i) the opportunity cost of placing task i before the first task j of a sequence is given by

$$\bar{c}_{d_o_j} = c_{d_o_j} - \pi_{d_i} + \pi_{o_j} = t_{ij} - cg - cg < 0; \quad (2.3)$$

- ii) the opportunity cost of placing task i before task j which actually follows task k in a sequence is given by

$$\bar{c}_{d_{o_j}} = c_{d_{o_j}} - \pi_{d_i} + \pi_{o_j} = t_{ij} - cg + cg - t_{kj} = t_{ij} - t_{kj}. \quad (2.4)$$

Hence the opportunity cost of placing a task before the first task of any sequence is always negative. Otherwise, the opportunity cost of placing a task i before a task j which actually follows task k in a sequence is negative if the deadheading time from EL_i to SL_j is smaller than the actual time from EL_k to SL_j .

2.3. Another opportunity cost

An alternate formulation of the problem generates different opportunity costs. For instance consider the following precedence graph $(N, \bar{A})$ where $\bar{A} = A \cup \{(o_i, d_i) : 1 \leq i \leq n\}$. The graph of the example in Section 2.1 is illustrated in Figure 2.3.

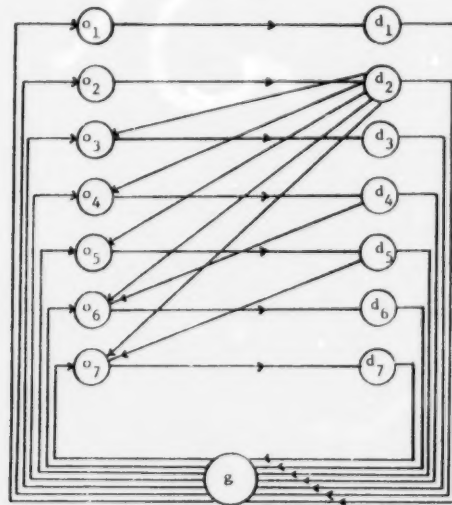


FIGURE 2.3

PRECEDENCE GRAPH $(N, \bar{A})$

The problem can be formulated as a minimum cost flow problem defined on the precedence graph $(N, \bar{A})$

$$\begin{aligned}
 & \text{Min} \quad \sum_{(i,j) \in \bar{A}} c_{ij} x_{ij} \\
 (\overline{FP}) \quad & \text{Subject to} \quad \sum_{j \in P_{d_i}} x_{d_i j} - x_{o d_i} = 0 \quad 1 \leq i \leq n \\
 & \quad \quad \quad x_{o d_i} - \sum_{j \in B_{o_i}} x_{j o_i} = 0 \quad 1 \leq i \leq n \\
 & \quad \quad \quad \sum_{i=1}^n (x_{g o_i} - x_{d g}) = 0 \\
 & \quad \quad \quad 0 \leq x_{ij} \leq 1 \quad (i,j) \in A \\
 & \quad \quad \quad 1 \leq x_{o d_i} \leq 1 \quad 1 \leq i \leq n
 \end{aligned}$$

where $c_{o d_i} = 0$, $1 \leq i \leq n$. The values of the variables in an optimal solution have the same interpretation as in the preceding model. Furthermore, note that $x_{o d_i} = 1$ in any feasible solution.

With this formulation, opportunity costs can be specified in terms of vehicle cost c_g and the cost of an alternate sequence of tasks. Indeed, given any set of optimal sequences of tasks, the following basic optimal solution of $(\overline{FP})$ is generated using the variables $x_{o d_i}$, $1 \leq i \leq n$, $x_{g o_i}$ for all first tasks i in any sequence, and $x_{d o_j}$ for all pairs of linked tasks i, j .

The basic solution for the example of Section 2.2 is illustrated in Figure 2.4

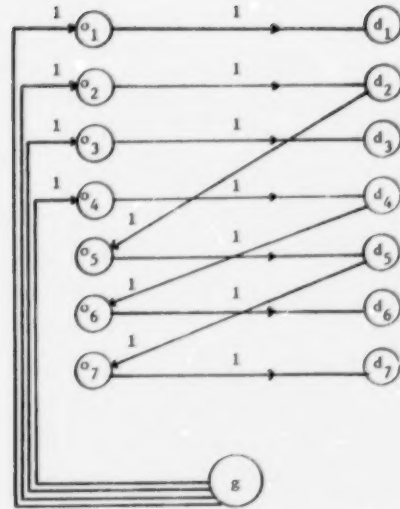


FIGURE 2.4

Basic optimal solution of $(\overline{FP})$

It is easy to verify that the simplex multipliers associated with this basis can be generated as follows:

- i) $\pi_g = 0$
- ii) $\pi_{o_i} = \pi_{d_i} \quad , \quad 1 \leq i \leq n$ (2.5)
- iii) $\pi_{o_i} = -cg \quad , \quad \text{if } i \text{ is the first task of any sequence}$
- iv) $\pi_{o_j} = \pi_{d_i} - t_{ij} \quad , \quad \text{if task } j \text{ follows task } i \text{ in any sequence.}$

Hence it follows from (2.5) that for all $1 \leq i \leq n$

$$\pi_{o_i} = \pi_{d_i} = -cg - DH_i \tag{2.6}$$

where DH_i is the deadheading time cumulated in the sequence up to task i . In the example of

Figure 2.4, $\pi_{o_3} = \pi_{d_3} = -cg - t_{25}$ and $\pi_{o_7} = \pi_{d_7} = -cg - t_{25} - t_{57}$. Of course, if i is the first task of the sequence, $DH_i = 0$.

Now, if task j could be linked to i , it follows from (2.6) that the opportunity cost is given by

$$\bar{c}_{d_{oj}} = c_{d_{oj}} - \pi_{d_i} + \pi_{o_j} = t_{ij} + DH_i - DH_j. \quad (2.7)$$

Hence if the deadheading time when j is executed after i ($DH_i + t_{ij}$) is smaller than the actual time (DH_j) then the opportunity cost $\bar{c}_{d_{oj}} < 0$.

3. Vehicle scheduling with time windows

The procedure for treating this problem solves a sequence of problems with fixed starting times which are modified according to the opportunity costs. Then, in the next section, the method will be extended to the problem with sliding time windows.

The vehicle scheduling problem with time windows can be formulated as follows [3, 7]:

$$\begin{aligned}
 & \text{Min} \quad \sum_{(i,j) \in AW} c_{ij} x_{ij} \\
 (WP) \quad & \text{Subject to} \quad \sum_{j \in PW_i} x_{ij} = 1 & 1 \leq i \leq n \\
 & \sum_{j \in BW_i} x_{ji} = 1 & 1 \leq i \leq n \\
 & x_{ij} = 0 \text{ or } 1 & (i,j) \in AW \\
 & x_{ij} > 0 \Rightarrow ET_i + t_{ij} \leq ST_j & (i,j) \in \overline{AW} \\
 & a_i \leq ST_i \leq b_i & 1 \leq i \leq n
 \end{aligned}$$

where $AW = \{(i, j): a_i + D_i + t_{ij} \leq b_j, 1 \leq i, j \leq n\} \cup \{(i, g), (g, i): 1 \leq i \leq n\}$,
 $\overline{AW} = \{(i, j) \in AW: i, j \neq g\}$, $PW_i = \{j: (i, j) \in AW\}$, $BW_i = \{j: (j, i) \in AW\}$, g denotes the garage
 where vehicles start and end their routes, and for all $(i, j) \in AW$

$$c_{ij} = \begin{cases} cg & \text{if } i=g \text{ or } j=g \\ t_{ij} & \text{otherwise} \end{cases}$$

Note that there are two sets of variables in WP : $x_{ij}, (i, j) \in AW$ and $ST_i, 1 \leq i \leq n$. Note also that once the variables ST_i are fixed, (WP) reduces to a problem with fixed starting times and can be reformulated in the forms (FP) or ($\overline{FP}$). Hence given an optimal solution for specified starting times, the procedure is to modify these to allow pairs of tasks to be linked. However, these modifications are selected to preserve the sequences in the optimal solution for the problem on hand. Hence this is a feasible solution for the new problem for which the optimal value can only decrease.

Assume that i and j belong to different sequences in some feasible solution to (WP) and j cannot be linked to i according to the values of ST_i and ST_j on hand; i.e., $ET_i + t_{ij} > ST_j$. Denote by S_i and S_j the sequences including i and j , respectively. If task j is to be linked to i , then i should be completed earlier at $ET_i - \Delta_i^-$ and task j should start later at $ST_j + \Delta_j^+$ such that

$$ET_i - \Delta_i^- + t_{ij} \leq ST_j + \Delta_j^+$$

or

$$\Delta_i^- + \Delta_j^+ \geq (ET_i + t_{ij}) - ST_j > 0. \quad (3.1)$$

Of course if S_i and S_j are to be preserved, then Δ_i^- and Δ_j^+ depend on the adjustments allowed by the other tasks before i in S_i and after j in S_j . It is easy to see that the Δ_i^- and Δ_j^+ are computed recursively for the tasks in the sequences as follows

$$\Delta_i^- = \begin{cases} ST_i - a_i & \text{if } i \text{ is the first task of } S_i \\ ST_i - \text{Max } (a_i, ET_{\mu_i} - \Delta_{\mu_i}^- + t_{\mu_i}) & \text{if } \mu_i \text{ is the predecessor of } i \text{ in } S_i ; \end{cases} \quad (3.2)$$

$$\Delta_i^+ = \begin{cases} b_i - ST_i & \text{if } i \text{ is the last task of } S_i , \\ \text{Min } (b_i, ST_{v_i} + \Delta_{v_i}^+ - t_{v_i} - D_i) - ST_i & \text{if } v_i \text{ is the successor of } i \text{ in } S_i . \end{cases} \quad (3.3)$$

A heuristic procedure ATW to handle problem (WP) can be derived. It starts with a set of starting times ST feasible for (WP). Given an optimal solution for the associated problem with fixed starting times, opportunity costs are computed according to any of the strategies developed in Section 2. Then the starting times are modified to allow some new pairs of tasks i, j with negative opportunity costs to be linked. Furthermore, these modifications preserve the optimal sequences for the problem on hand. The procedure is repeated a maximal number of times, or it stops when all pairs (i, j) with negative arc costs cannot be linked using the modifications. The steps of the procedures are specified in the Appendix.

4. Vehicle scheduling with sliding time windows

The results of the preceding section are now extended to handle the problem with sliding time windows. This problem can be formulated mathematically as follows:

$$\begin{aligned} & \text{Min} \quad \sum_{(i,j) \in AS} c_{ij} x_{ij} \\ (SP) \quad \text{Subject to} \quad & \sum_{j \in PS_i} x_{ij} = 1 \quad 1 \leq i \leq n \\ & \sum_{j \in BS_i} x_{ji} = 1 \quad 1 \leq i \leq n \end{aligned}$$

$$x_{ij} = 0 \text{ or } 1 \quad (i, j) \in AS$$

$$x_{ij} > 0 \Rightarrow ET_i + t_{ij} \leq ST_j \quad (i, j) \in \overline{AS}$$

$$\bar{a}_k + f_i \leq ST_i \leq \bar{a}_k + f_i + w_i \quad i \in T_k, \quad 1 \leq k \leq K$$

$$\bar{a}_k^1 \leq \bar{a}_k \leq \bar{a}_k^2 \quad 1 \leq k \leq K.$$

For $1 \leq i \leq n$, $w_i \geq 0$ denotes the length of the time window associated with task i , and $k(i)$ denotes the index k of the set T_k to which i belongs. Furthermore, it is assumed that $f_i \geq 0$. Also the set of admissible arcs $AS = \overline{AS} \cup \{(i, g), (g, i): 1 \leq i \leq n\}$ where for all $1 \leq i \neq j \leq n$, $(i, j) \in \overline{AS}$ if and only if

$$f_i + D_i + t_{ij} \leq f_j + w_j \text{ when } k(i) = k(j)$$

or

$$\bar{a}_{k(i)}^1 + f_i + D_i + t_{ij} \leq \bar{a}_{k(j)}^2 + f_j + w_j \text{ when } k(i) \neq k(j).$$

Finally $PS_i = \{j: (i, j) \in AS\}$, $BS_i = \{j: (j, i) \in AS\}$, and g denotes a garage where vehicles start and end. The costs c_{ij} are defined as in (WP).

Now, there are three sets of variables in (SP): x_{ij} , $(i, j) \in AS$, ST_i , $1 \leq i \leq n$, and $\bar{a}_k$, $1 \leq k \leq K$. Once variables $\bar{a}_k$ are fixed, (SP) reduces to a problem (WP) with time windows. Hence, by analogy with the preceding section, the procedure for treating (SP) solves a sequence of problems with time windows by modifying the values of variables $\bar{a}_k$ according to opportunity costs to introduce new potential links between tasks.

As we shall see, it is more complex to specify modifications to preserve the sequences in the optimal solution on hand. Indeed if for some $1 \leq k \leq K$, $\bar{a}_k$ is modified, then

all tasks $i \in T_k$ are influenced.

To simplify the notation, let $\bar{a}_{k(i)} + f_i = a_i$ and $\bar{a}_{k(i)} + f_i + w_i = b_i$. Denote by S_i the sequence including i , and by M_i the set including i and the tasks in S_i completed before i . Furthermore, assume that each task in M_i belongs to a different set T_k . If for all $k \neq k(i)$, $\bar{a}_k$ remains unchanged whenever $M_i \cap T_k \neq \emptyset$, then the earliest time $EH(i)$ at which the vehicle is available to execute task i is

$$EH(i) = \begin{cases} -\infty & , \text{ if } i \text{ is the first task of } S_i \\ ET_{\mu_i} + t_{\mu_i} - \Delta_{\mu_i}^- & , \text{ if } \mu_i \text{ is the predecessor of } i \text{ in } S_i . \end{cases} \quad (4.1)$$

Hence, if the sequence S_i is to be preserved, then the upperbound b'_i of the adjusted time window of i after modifying $\bar{a}_{k(i)}$ should satisfy

$$b'_i \geq EH(i) .$$

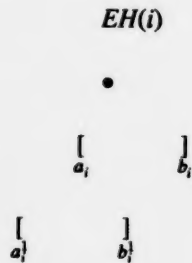


FIGURE 4.1

Sliding window

It follows that the amount of time $\Delta A_{k(i)}^-$ by which $\bar{a}_{k(i)}$ can be decreased is bounded by $b_i - EH(i)$; i.e.

$$\Delta A_{k(i)}^- \leq b_i - EH(i) . \quad (4.2)$$

As a consequence we define

$$\Delta A_k^- = \min \{ \bar{a}_k - \bar{a}_k^1, \min_{i \in T_k} \{ b_i - EH(i) \} \} . \quad (4.3)$$

By analogy, we determine ΔA_k^+ , the amount of time by which $\bar{a}_k$ can be increased. Denote by N_i the set including i and the tasks in S_i completed after i . Assume also that each task in N_i belongs to a different set T_k . If for all $k \neq k(i)$, $\bar{a}_k$ remains unchanged whenever $N_i \cap T_k \neq \emptyset$, then the latest time $LH(i)$ at which the vehicle is available to execute task i is

$$LH(i) = \begin{cases} +\infty & , \text{ if } i \text{ is the last task in } S_i \\ ST_{v_i} - t_{iv_i} + \Delta v_i^+ - D_i & , \text{ if } v_i \text{ is the predecessor of } i \text{ in } S_i . \end{cases} \quad (4.4)$$

It follows that the amount of time ΔA_k^+ by which $\bar{a}_k$ can be increased is

$$\Delta A_k^+ = \min \{ \bar{a}_k^2 - \bar{a}_k, \min_{i \in T_k} \{ LH(i) - a_i \} \} \quad (4.5)$$

Note that other definitions of ΔA_k^- and ΔA_k^+ can be given, but using (4.3) and (4.5) it can be shown that the sequences remain feasible if the modifications are all of the same type (i.e. some $\bar{a}_k$ are increased or some $\bar{a}_k$ are decreased but not both). A proof of this result is given in the Appendix.

By analogy with the preceding section and referring to relations (3.2) and (3.3), the maximal variations $\bar{\Delta}_i^-$ and $\bar{\Delta}_i^+$ can be computed recursively to preserve the sequences on hand:

$$\bar{\Delta}_i^- = \begin{cases} ST_i - (a_i - \Delta A_{k(i)}^-) & \text{if } i \text{ is the first task of } S_i \\ ST_i - \max \{ a_i - \Delta A_{k(i)}^-, ET_{\mu_i} - \bar{\Delta}_{\mu_i}^- + t_{\mu_i i} \} & \text{if } \mu_i \text{ is the predecessor of } i \text{ in } S_i \end{cases} \quad (4.6)$$

$$\tilde{\Delta}_i^+ = \begin{cases} b_i + \Delta A_{k(i)}^+ - ST_i & \text{if } i \text{ is the last task of } S_i \\ \min(b_i + \Delta A_{k(i)}^+, ST_{v_i} + \tilde{\Delta}_{v_i}^+ - t_{iv_i} - D_i) - ST_i & \text{if } v_i \text{ is the successor of } i \text{ in } S_i. \end{cases} \quad (4.7)$$

Finally the tasks i and j form a candidate pair to be linked only if their sets S_i and S_j verify

$$\left\{ \bigcup_{l \in M_i} k(l) \right\} \cap \left\{ \bigcup_{l \in N_j} k(l) \right\} = \emptyset \quad (4.8)$$

Hence, if we apply modifications of type $\tilde{\Delta}^-$ to task $l \in M_i$, then modifications of type Δ^+ can be applied to tasks $l \in N_j$ because their time windows are not affected by the preceding modifications of type $\tilde{\Delta}^-$.

A heuristic procedure ASW to handle problem (SP) can be derived. It starts with a feasible solution $\bar{a}$ for (SP). Given an optimal solution for the associated (WP) problem, opportunity costs are computed according to any of the strategies developed Section 2. Then among all pairs of tasks (i, j) which can be linked after modifications, the one with smallest negative opportunity cost is selected. Then either some variables $\bar{a}_k$ are increased or some variables $\bar{a}_k$ are decreased, and the starting times are adjusted to allow j to be linked to i . Furthermore these modifications preserve the optimal sequences for the problem on hand. The procedure is repeated by solving this new (WP) problem. The procedure stops after a specified number of iterations or when all pairs with negative opportunity costs cannot be linked through the modifications of variables $\bar{a}$ and ST known to preserve the optimal sequences on hand. The steps of the procedure are specified in the Appendix.

A variant of this procedure can be specified where the variables $\bar{a}_k$ are modified by increasing some and decreasing others. We shall refer to this variant as procedure ASW2. In this variant the sequences on hand do not necessarily remain feasible for the new (WP)

problem. Consequently, the value of the objective function is not necessarily monotone during the procedure. Hence, if the value of the objective increases, another pair (i, j) with negative opportunity cost is selected to repeat the procedure rather than to compute new opportunity costs associated with the optimal solution for the (WP) problem on hand.

5. Numerical results

These algorithms were implemented in FORTRAN on the CYBER 173 of the University of Montreal. The problems used to test and compare the results for different opportunity costs are summarized in TABLE 5.1. These are school busing problems already introduced in [3, 4, 9, 10].

Problems	n	k	AWI	Average D_i (Sec.)	Average w_i (Sec.)	$\bar{a}_k^2 - \bar{a}_k^1$ (Sec.)
48 AM	48	6	751	18.7	4.6	10
48 PM	48	6	789	18.7	4.6	10
48 AM+5	48	6	827	18.7	9.6	10
48 PM+5	48	6	844	18.7	9.6	10
100 AM	100	46	1793	34.4	7.1	10
100 PM	100	46	2797	34.4	7.1	10
100 AM+5	100	46	2093	34.4	12.1	10
100 PM+5	100	46	3018	34.4	12.1	10
151 AM	151	55	4311	32.9	7.8	10
151 PM	151	55	6529	32.9	7.8	10
151 AM+5	151	55	5108	32.9	12.8	10
151 PM+5	151	55	7077	32.9	12.8	10
223 AM	223	65	10031	32.6	7.8	10
223 PM	223	65	14487	32.6	7.8	10
223 AM+5	223	65	11823	32.6	12.8	10
223 PM+5	223	65	15695	32.6	12.8	10

TABLE 5.1

The value of $|AW|$ in Table 5.1 represents the number of arcs in the set AW for this problem (WP). The initial solution for the algorithm (ASW) is obtained by selecting $\bar{a}_k = \frac{\bar{a}_k^1 + \bar{a}_k^2}{2}$ for $1 \leq k \leq K$. Then, the initial values for variables in ST are selected using heuristic procedures introduced in [10] to solve the associated time window problem (WP). Furthermore, the computer code RNET is used to solve problems with fixed starting times, and the multipliers generated by this procedure are used to compute the first type of opportunity costs.

In the implementation of the algorithms (as specified in the Appendix) used in our tests, the following values of the parameters are selected:

- ρ_1 in ATW = 33 (maximal number of arcs introduced)
- ρ_2 in ATW = 14 (number of iterations)
- ρ_3 in ASW = k (number of iterations)
- ρ_3 in ASW2 = $k/2+5$ (number of iterations) .

To reduce the execution time an additional stopping criterion for the algorithm ATW was used: if the value of the objective function remains unchanged, the procedure stops.

Results in Table 5.2 are generated by applying algorithm ASW2 after algorithm ASW. They indicate the number of vehicles required in the solutions generated by using different opportunity costs:

- . Strategy A : opportunity costs computed with RNET optimal multipliers;
- . Strategy B : opportunity costs as specified in section 2.2;
- . Strategy C : opportunity costs as specified in section 2.3.

The results indicate that strategy A uniformly dominates the others. Hence the results analyzed in the remaining of this section are obtained using this strategy.

Problems	Strategy			Best known solution for $\bar{a}_k = \frac{\bar{a}_k^1 + \bar{a}_k^2}{2}$	% improvement
	A	B	C		
48 AM	16	16	16	17	5.88
48 Pm	15	15	15	17	11.76
48 AM+5	15	15	16	15	0
48 PM+5	14	14	14	14	0
100 AM	56	57	61	63	11.11
100 PM	46	47	48	51	9.80
100 AM+5	53	56	56	59	10.16
100 PM+5	44	45	45	46	8.69
151 AM	87	89	89	94	7.45
151 PM	70	70	71	75	6.66
151 AM+5	84	84	86	88	4.55
151 PM+5	67	68	67	70	4.29
223 AM	130	133	134	134	2.98
223 PM	98	99	98	107	8.41
223 AM+5	115	123	126	126	8.73
223 PM+5	92	92	92	98	6.12

TABLE 5.2

In Table 5.2, the best known solution for $\bar{a}_k = \frac{\bar{a}_k^1 + \bar{a}_k^2}{2}$ was generated with exact type procedure to solve (WP) as presented in [3, 4]. Finally, the last column of the table indicate the percentage of improvement over the best known solution by allowing flexibility on the values of $\bar{a}_k$ and using strategy A to define the opportunity costs.

The results of Table 5.3 compare the results using two different initial solutions and indicate the benefits of applying algorithm ASW2 after ASW rather than using only algorithm ATW with $\bar{a}_k = \frac{\bar{a}_k^1 + \bar{a}_k^2}{2}$. The results show that there is always a gain in exploiting the flexibility on the values of $\bar{a}_k$. Furthermore, the initial solution influences the final one, but this influence is not uniform in the sense that a better final solution can be generated from a poor initial one.

Problems	Initial solution 1			Initial solution 2		
	initial	ATW	ASW+ ASW2	initial	ATW	ASW+ ASW2
48 AM	17	17	16	18	18	16
48 PM	17	17	15	17	17	15
48 AM+5	17	16	15	17	16	15
48 AM+5	17	15	14	14	14	14
100 AM	70	66	56	67	66	56
100 PM	53	51	46	53	53	47
100 AM+5	65	61	53	64	62	54
100 PM+5	53	47	44	49	47	43
151 AM	105	96	87	99	98	90
151 PM	81	77	70	80	77	70
151 AM+5	93	92	84	92	91	83
151 PM+5	81	75	67	74	72	65
223 AM	144	142	130	142	141	128
223 PM	116	112	98	117	113	98
223 AM+5	137	132	115	131	131	115
223 PM+5	105	100	92	104	101	91

TABLE 5.3

Finally, table 5.4 compares the relative efficiency of using algorithm ASW, or ASW2, or ASW2 after ASW. CPU times are also given. The results indicate that the strategy of using ASW2 after ASW always dominates the others and that the increase in CPU times is not very important.

To summarize, the numerical results seem to indicate an improvement in the number of vehicles required if we allow flexibility on the values of $\bar{a}_k$ and the best solutions are obtained using strategy A to compute the opportunity costs.

Problems	Number of vehicles			CPU time (Sec)		
	ASW	ASW2	ASW+ ASW2	ASW	ASW2	ASW+ ASW2
48 AM	16	16	16	11.55	12.53	12.03
48 PM	15	15	15	13.52	12.88	14.07
48 AM+5	15	16	15	13.29	15.01	13.75
48 PM+5	14	14	14	15.62	13.04	15.85
100 AM	56	57	56	70.21	67.39	71.32
100 PM	47	47	47	74.60	56.69	75.44
100 AM+5	54	55	54	90.87	68.66	92.58
100 PM+5	43	44	43	75.78	66.99	77.13
151 AM	90	91	90	176.71	161.52	191.43
151 PM	70	70	70	225.63	201.10	227.79
151 AM+5	83	83	83	273.88	210.05	274.90
151 PM+5	65	65	65	269.92	237.14	270.44
223 AM	128	129	128	529.10	409.01	532.09
223 PM	98	101	98	603.52	453.48	604.78
223 AM+5	115	116	115	845.30	615.88	849.99
223 PM+5	92	95	91	567.55	443.04	644.74

TABLE 5.4

APPENDIX

This appendix includes detailed versions of procedures ATW and ASW. Furthermore, in Section A.3 we verify that the modifications used in ASW preserve the optimal sequences for the problem (WP) on hand.

A.1 Algorithm time window (ATW)

Step 0 Assume that ST is a set of starting times for (WP) and x is an optimal solution for the associated fixed starting times problem. Let $\bar{x} = x$, ρ_1 be the maximal number of new pairs i, j allowed to be linked at each iteration, and ρ_2 the maximal number of iterations. Let $\text{count} = 0$.

Step 1 Let $\text{count} = \text{count} + 1$.

Determine the set X_1 of pairs of tasks (i, j) such that

- i) i and j belong to different sequences;
- ii) the opportunity cost of linking j to i is negative;
- iii) $\Delta_i^- + \Delta_j^+ \geq (ET_i + t_{ij}) - ST_j$.

If $X_1 = \emptyset$, $\bar{x}$ is best solution generated, and STOP.

Step 2 Let $F(i) = 1$, $1 \leq i \leq n$. $X_2 = \emptyset$, $\text{cou} = 0$.

2.1. Let (i, j) be the element of X_1 with the smallest $\bar{c}_{ij}$. $X_1 = X_1 - \{(i, j)\}$.

2.2. If $F(i) = F(j) = 1$, then $X_2 = X_2 \cup \{(i, j)\}$, $F(k) = 0$ for all tasks k in the sequences to which i and j belongs, and $\text{cou} = \text{cou} + 1$.

If $\text{cou} < \rho_1$, and if $X_1 \neq \emptyset$, return to 2.1.

Step 3 For each pair $(i, j) \in X_2$, j can be limited to i as follows:

i) Determine

$$\bar{\Delta}_i^- = \left[\frac{\Delta_i^-}{\Delta_i^- + \Delta_j^+} \right] (ET_i + t_{ij} - ST_j)$$

$$\bar{\Delta}_j^+ = \left[\frac{\Delta_j^+}{\Delta_i^- + \Delta_j^+} \right] (ET_i + t_{ij} - ST_j)$$

ii) For $l \in M_i$

$$ST_l = ST_l - \bar{\Delta}_i^-.$$

iii) For $l \in N_j$

$$ST_l = ST_l + \bar{\Delta}_j^+.$$

Step 4 Generate an optimal solution x for the problem with these new starting times, and let

$$\bar{x} = x.$$

If $\text{count} < \rho_2$, return to Step 1. Otherwise, $\bar{x}$ is the best solution generated, and

STOP. □

To compute the opportunity costs in Step 1, any of the relations (2.2), (2.4), or (2.7), or any of their combinations can be used. In Step 2 where the set of pairs which can be linked is identified, the parameters $F(i)$ are used to reduce to one the maximal number of modification in any sequence.

Finally, in Step 3 rather than adjusting the starting times with the quantities Δ_i^- and Δ_j^+ we prefer to make minimal adjustments ($\bar{\Delta}_i^-$ and $\bar{\Delta}_j^+$) allowing j to be linked to i . Furthermore, the minimal adjustments are redistributed according to the flexibilities (see relations specify in $\bar{\Delta}_i^-$, $\bar{\Delta}_j^+$).

A.2 Algorithm sliding window (ASW)

Step 0 Assume that $\bar{\alpha}$ and ST is feasible for (SP) and denote by x an optimal solution for the associated fixed starting times problem. Let $\bar{x} = x$, and let p_3 be the maximal number of iterations. Let $\text{cnt} = 0$.

Step 1 Let $\text{cnt} = \text{cnt} + 1$.

Determine the set XA_1 of pairs of tasks (i, j) such that

- i) i and j belong to different sequences;
- ii) $\{ \bigcup_{l \in M_i} k(l) \} \cap \{ \bigcup_{l \in N_j} k(l) \} = \emptyset$;
- iii) the opportunity cost of linking j to i is negative;
- iv) $\bar{\Delta}_i^- + \Delta_j^+ \geq (ET_i + t_{ij}) - ST_j$ or $\Delta_i^- + \bar{\Delta}_j^+ \geq (ET_i + t_{ij}) - ST_j$

If $XA_1 = \emptyset$, $\bar{x}$ is the best solution generated, and STOP.

Step 2 Select the pair $(i, j) \in XA_1$ with the smallest $\bar{c}_{ij}$. If $\bar{\Delta}_i^- + \bar{\Delta}_j^+ \geq (ET_i + t_{ij}) - ST_j$, then go to STEP 4.

Step 3 Modify the starting times and slide time windows to allow j to be linked to i :

i) $\bar{a}_{k(l)} = \bar{a}_{k(l)} - \Delta A_{k(l)}^-$, $l \in M_i$.

ii) For sequence S_i modify the starting times

$$ST_l = ST_l - \tilde{\Delta}_i^- , l \in M_i .$$

iii) For sequence S_j modify the starting times

$$ST_l = ST_l + \Delta_j^+ , l \in N_j .$$

iv) For any other sequence S , if $\bar{a}_{k(l)}$ is not modified for any $l \in S$, then the starting times are not modified. Otherwise, let r denotes the latest completed task of S for which $\bar{a}_{k(r)}$ is modified. Then for all $l \in M_r$,

$$ST_l = \text{Max}\{EH(l), a'_l\} .$$

(See the notation introduce in Theorem A.1).

Step 4 Modify the starting times and slide time windows to allow j to be linked to i :

i) $\bar{a}_{k(l)} = \bar{a}_{k(l)} + \Delta A_{k(l)}^+$, $l \in N_j$.

ii) For sequence S_j modify the starting times

$$ST_l = ST_l + \tilde{\Delta}_j^+ , l \in N_j .$$

iii) For sequence S_i modify the starting times

$$ST_l = ST_l - \Delta_i^- , l \in M_i .$$

iv) For any other sequence S , if $\bar{a}_{k(l)}$ is not modified for any $l \in S$, then the starting times are not modified. Otherwise let r denotes the earliest completed task of S for which $\bar{a}_{k(r)}$ is modified. Then for all $l \in N_r$,

$$ST_l = \text{Min}\{LH(l), b'_l\} .$$

Step 5 Generate an optimal (or a good) solution x for the new problem (WP), and let $\bar{x} = x$.

If $\text{cnt} < \rho_3$, return to Step 1. Otherwise, $\bar{x}$ is the best solution generated, and STOP. □

As before, the opportunity costs in Step 1 can be computed according to (2.2), (2.4), or (2.7), or any combination of these. It is implicitly assumed that a better solution cannot be obtained with the time windows available. Algorithm Time Window (ATW) or any algorithm in [2, 3, 4, 7] can be used in Step 5.

The algorithm for procedure ASW2 is quite similar to the preceding. The minor adjustments are summarized as follows:

- . the test in iv) of Step 1 becomes

$$\bar{\Delta}_i^- + \bar{\Delta}_j^+ \geq (ET_i + t_{ij}) - ST_j ;$$

- . Steps 3 and 4 are merged to modify accordingly variables $\bar{a}_k$ and the starting times.

A.3 Preserving optimal sequences

We verify that the sequences remain feasible if the values of some $\bar{a}_k$ are decreased. A similar result can be derived if the values of some $\bar{a}_k$ are increased.

Theorem A.1 Suppose that $\bar{a}_k$ is modified to $\bar{a}_k - \Delta A_k^-$ for all $k \in \Gamma \subset \{1, 2, \dots, K\}$, then the starting time of the tasks can be adjusted to preserve the sequences in the solution on hand before the modifications.

Proof

For $1 \leq i \leq n$,

$$a'_i = \begin{cases} a_i - \Delta A_{k(i)}^- & \text{if } k(i) \in \Gamma \\ a_i & \text{otherwise} \end{cases}$$

and

$$b'_i = \begin{cases} b_i - \Delta A_{k(i)}^- & \text{if } k(i) \in \Gamma \\ b_i & \text{otherwise} \end{cases}$$

Consider any sequence S in the optimal solution on hand which includes at least on task l with a modified $\bar{a}_{k(l)}$. Let r denotes the latest completed task of S for which $\bar{a}_{k(r)}$ is modified; i.e., $\bar{a}_{k(l)}$ is not modified for all $l \in N_r$, $l \neq r$ (where N_r denotes the set including r and all tasks in S completed after r). Then specify the new starting time for the tasks in S as follows:

$$ST'_l = \begin{cases} \text{Max } \{EH(l), a'_l\} & \text{if } l \in M_r \\ ST_l & \text{otherwise} \end{cases} \quad (A.1)$$

where M_r denotes the set including r and all tasks in S completed before r). We now show that the sequence S remains feasible for the new starting times according to the new time windows.

For $l \in N_r$, $l \neq r$, $ST'_l = ST_l \in [a_l, b_l] = [a'_l, b'_l]$. But for $l \in M_r$, $ST'_l = \text{Max } \{EH(l), a'_l\} \geq a'_l$. If $ST'_l = a'_l$, then $ST'_l \in [a'_l, b'_l]$. Otherwise,

$$ST'_l = EH(l) = b_l - [b_l - EH(l)] \leq b_l - \Delta A_{k(l)}^- = b'_l,$$

and $ST'_l \in [a'_l, b'_l]$.

For any task l in S , denote μ_l its predecessor in S (unless l is the first task in S) and v_l its successor in S (unless l is the last task in S). Now, the relation $ET'_l + t_{\mu_l} \leq ST'_{v_l}$ is verified

trivially for $l \in N_r$, $l \neq r$, and l is not the last task in S . Indeed for these tasks l , $ST'_l = ST_l$.

To analyse the case where $l \in M_r$, $l \neq r$, first note that referring to (3.2)

$$ST_l - \Delta_l^- = \text{Max}\{a_l, ET_{\mu_l} - \Delta_{\mu_l}^- + t_{\mu_l}\}. \quad (\text{A.2})$$

Since $a'_l \leq a_l$, then

$$ST_l - \Delta_l^- \geq \text{Max}\{a'_l, ET_{\mu_l} - \Delta_{\mu_l}^- + t_{\mu_l}\} = \text{Max}\{a'_l, EH(l)\} = ST'_l. \quad (\text{A.3})$$

Now, from (A.1)

$$ST'_{v_l} \geq EH(v_l) = ET_l + t_{N_l} - \Delta_l^- = ST_l - \Delta_l^- + D_l + t_{N_l}, \quad (\text{A.4})$$

and substituting (A.3) in (A.4)

$$ST'_{v_l} \geq ST'_l + D_l + t_{N_l} = ET'_l + t_{N_l}.$$

To complete the proof, we have to verify that

$$ET'_r + t_{N_r} \leq ST'_{v_r} = ST_{v_r}. \quad (\text{A.5})$$

Two cases have to be considered. Either $ST'_r = a'_r$, and then (A.5) is verified since $ST'_r \leq ST_r$ implies that $ET'_r \leq ET_r$. Or $ST'_r = EH(r) = ET_{\mu_r} + t_{\mu_r} - \Delta_{\mu_r}^-$. If $EH(r) \leq a_r \leq ST_r$, then (A.5) is verified as above. Otherwise, referring to (A.2), equality holds in (A.3) and

$$ST_r - \Delta_r^- = ST'_r.$$

Then

$$ST'_{v_r} \geq ET_r + t_{N_r} \geq ET_r - \Delta_r^- + t_{N_r} = ET'_r + t_{N_r},$$

since $\Delta_r^- \geq 0$. □

References

- [1] Desrosiers, J., "Le transport scolaire: les horaires d'écoles et les circuits d'autobus", Publication #140, Centre de recherche sur les transports, Université de Montréal, Canada, 1979.
- [2] Desrosiers, J., Ferland, J.A., Rousseau, J.-M., Lapalme, G., Chapleau, L., "TRANSCOL: a multi-period school bus routing and scheduling systems," in: A.J. Swersey, E.J. Ignall (ed.), Studies in the Management Sciences 22, North-Holland, Amsterdam, 1986, 47-71.
- [3] Desrosiers, J., Soumis, F., Desrochers, M., "Routing with time windows by column generation", Networks (1984) 545-565.
- [4] Desrosiers, J., Soumis, F., Desrochers, M., Sauvé, M., "Routing and scheduling with time windows solved by network relaxation and branch-and-bound on time variables", in: J.-M. Rousseau (ed.), Computer scheduling of public transport II, North-Holland, Amsterdam, 1985, 451-471.
- [5] Desrosiers, J., Soumis, F., Sauvé, M., "Lagrangian relaxation for routing with time windows", in: J.P. Brans (ed.), Operational Research, North-Holland, Amsterdam, 1984, 673-684.
- [6] Desrosiers, J., Soumis, F., Desrochers, M., Sauvé, M., "Methods for routing with time windows", European Journal of Operational Research 23 (1986), 236-245.
- [7] Ferland, J.A., "The single depot vehicle scheduling problem", Publication #385, Centre de recherche sur les transports, Université de Montréal, Canada, 1985.

- [8] Ferland, J.A., "Programmation linéaire", Notes de cours, Département d'informatique et de recherche opérationnelle, Université de Montréal, Canada (revised version), 1985.

- [9] Fortin, L., "Horaires d'autobus scolaires avec fenêtre de temps sur les horaires d'écoles", Master thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal, Canada, 1986.

- [10] Langlet, M.F., "Approches heuristiques au problème d'horaires avec fenêtres de temps", Master report, Département d'informatique et de recherche opérationnelle, Université de Montréal, Canada, 1984.

- [11] Luenberger, D.G., " Linear and nonlinear programming ", Addison Wesley, Reading, 1984.